Design and Analysis of Algorithms Assignment 3

Harrison Lee, Alex Zhao

February 7, 2019

Question 1. *(6.5) String Segmentation (Segmented Least Squares Fit? As from class.)*

Given: Given a string of letters $y = y_1 y_2 ... y_n$ with a length of n. We can assume we can get the quality of any string using the function $quality(x)$ for string x. Total quality of a segmentation is the qualities of each of its blocks.

Find: Find an efficient algorithm the gets the segmentation of maximum total quality.

Algorithm 1. *Find the optimal string segmentation.*

Proof. Subproblems: $OPT(j)$, is the total quality of the optimal segmenation from character 1 to character j. $Segment(i, j)$ is the segmentation from i to j.

Recurrence: $OPT(j) = max\{OPT(i) + quality(Segment(i + 1, j))$ for all i from 1 to $j - 1$

This is similar to the answer for Segmented Least Squares Fit, as discussed in lecture. The main change needed is to replace the error function with the quality function in this problem and to maximize quality rather than minimize error.

Full Algorithm:

```
$OPT(1) = quality(x_1)$    // Base Case
$segmentations = [[1]]$    // List of list of segmentation break points for each $OPT(j)$
for j = 2, j $\leq$ n, j++
  currentQuality = $quality(Segment(1, j)$
  currentSegmentation = 1
  for i = 2, i ¡ j, i++    // Find $OPT(j)$
    if $OPT(i)$ + $quality(Segment(i + 1, j)$ $\geq$ currentQuality
      currentQuality = $OPT(i)$ + $quality(Segment(i + 1, j)$
      currentSegmentation = i
  $OPT(j)$ = currentQuality
  $segmentations[j] = segmentations[i] + currentSegmentation$
$return\ OPT[n], segmentations$
```

Running Time: $O(n^2)$. Each character is compared to the OPT values of all prior characters. □

Question 2. *(6.28) Scheduling ((un-)Weighted Interval Scheduling? Lecture 25Jan2019; nope, nvm. Job start times can be defined. More like knapsack then? Except limitations by deadline, added parameter to knapsack's size limits)*

Given: You have n weeks and s_i parts to be produced each of those weeks. You must decide between two shipping companies, Company A which charges $r * s$ to ship in a given week while Company B charges c each week in blocks of four consecutive weeks.

Find: A schedule deciding between company A or B for each of those n weeks while following company B's restrictions. Cost is the amount paid in shipping costs.

Give a polynomial time algorithm that takes a sequence of supply values, $s_1, s_2, \ldots, s_n$ and returns a schedule of minimum cost.

Algorithm 2. *Find the optimal schedule.*

Proof. Subproblems: $OPT(j)$, or the optimal schedule from week 0 to week j. This can be expanded from week 4 to week n.

Recurrence: $OPT(j) = min\{\begin{matrix} r * s_j + OPT(j-1) \\ 4 * c + OPT(j-4) \end{matrix}$, where $OPT(j)$ represents the optimal shipping costs possible from week 0 to week j. r, s, and c are as defined in the problem, referring to company A's cost per weight unit, total weight during week j, and company B's cost per week respectively.

Full Algorithm:

```
OPT[0, 1, 2, 3] = [0, $s_1 * r, s_1 * r + s_2 * r, s_1 * r + s_2 * r + s_3 * r$]
for j = 0, j ≤ n, j++
  if $OPT[j-4] + 4 * c < OPT[j-1] + r * s_j$
    $OPT[j] = OPT[j-4] + 4 * c$
  else:
    $OPT[j] = OPT[j-1] + r * s_j$
$returnOPT[n]$
```

Running Time: $O(n)$. Each week is traversed once, while two already calculated $OPT(j)$s are accessed, $OPT(j-1)$ and $OPT(j-4)$, each week.

□